A Hough Transform Approach to Solving Linear Min-Max Problems

Carmi Grushko
carmi.grushko@gmail.com

Technion, Israel Institute of Technology

March 4, 2013

Several ways to accelerate the solution of 2D/3D linear min-max problems in n constraints are discussed. We also present an algorithm for solving such problems in the 2D case, which is superior to CGAL's linear programming solver, both in performance and in stability.

1 Purpose

This work is focused on several ways to accelerate the solution of 2D/3D linear min-max problems in n constraints. We also present an algorithm for solving such problems in the 2D case, which is superior to CGAL's linear programming solver, both in performance and in stability.

Problem 1. Linear Min-Max problem, also known as L_∞ linear optimization

$$\arg\min_{x,y\in\mathbb{R}}\left(\max_{i\in[n]}|a_ix+b_iy+c_i|\right)=\arg\min_{x,y}\left(\left\|M\cdot(x,y,1)^T\right\|_\infty\right),$$

where $M=[a|b|c]$.

This problem can be re-written as a Linear Programming of the following form.

Problem 2.

$$\begin{aligned} \text{minimize}_{x,y,t} \quad & t \\ \text{s.t} \quad & a_1x+b_1y+c_1\leq t \\ & \vdots \\ & a_nx+b_ny+c_n\leq t. \end{aligned}$$

It is known for several decades (Megiddo (1984)) that general Linear Programming problems (and thus, also linear min-max problems) can be solved in $\mathcal{O}(n)$ time when the dimension is constant. Therefore, there is only hope to demonstrate a constant factor acceleration; however, in real applications, this can be valuable.

2 The Hough Transform

We begin with a quick overview of a natural extension to the Hough Transform Hough (1962) to 3D. We note that the following theorems, although proved for 3D, hold equally well in 2D; the proofs are identical if we rename the z coordinate into y.

Given a point $p=(a,b,c)$ in $\mathbb{R}^3$, we define its dual plane as $\mathcal{H}(p)=\ell(x,y)=ax+by-c$, and given a plane $\pi(x,y)=ax+by+c$, we define its dual point as $\mathcal{H}(\pi)=(a,b,-c)$. The usefulness of these definitions is highlighted in the following lemmas.

Lemma 3. *A point $p=(x_0,y_0,z_0)$ is above a plane $\pi(x,y)=ax+by+c$ iff the plane $\mathcal{H}(p)$ is below the point $\mathcal{H}(\pi)$. Moreover, p is on π iff $\mathcal{H}(\pi)$ is on $\mathcal{H}(p)$.*

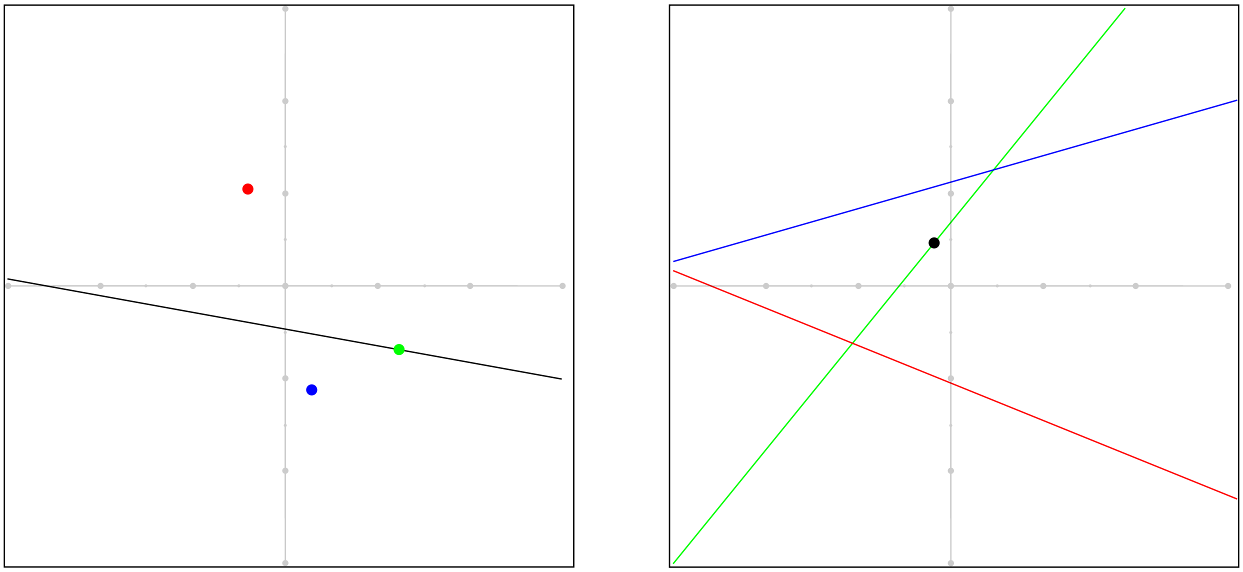

Figure 1: Hough Transform of three points and a line (plane in 3D). Lines and points, and their duals, are related by their color.

Proof. We know that $\pi(x_0, y_0) = x_0 a + y_0 b + c < z_0$; by definition of the Hough transform, $[\mathcal{H}(p)]\left(\mathcal{H}(\pi)_x, \mathcal{H}(\pi)_y\right) = x_0 \mathcal{H}(\pi)_x + y_0 \mathcal{H}(\pi)_y - z_0$ which equals $x_0 a + y_0 b - z_0$. Because $x_0 a + y_0 b + c < z_0$, we must have $x_0 a + y_0 b - z_0 < -c$, concluding that $[\mathcal{H}(p)]\left(\mathcal{H}(\pi)_x, \mathcal{H}(\pi)_y\right) < \mathcal{H}(\pi)_z$. A very similar argument can be used to show that a point is on a plane iff its dual is on its dual. □

Lemma 4. *The upper envelope of a set of planes corresponds to the lower convex hull of the planes' dual points.*

Proof. By definition, a point p on the upper envelope of a set of planes $\{\pi_i\}$ is above all of them (except several on which it lies), and by Lemma 3 we have that the plane $\mathcal{H}(p)$ is below all the points $\mathcal{H}(\pi_i)$ (except several which it touches). The converse is also true - every plane π that is part of the lower convex hull is lower than all points p_i (except several which it touches), therefore $\mathcal{H}(\pi)$ must be above all the planes $\mathcal{H}(p_i)$. □

We will henceforth denote the convex hull of a set of points P by $\mathcal{CH}(P)$, and its lower convex hull by $\mathcal{LH}(P)$. Figure 1 demonstrates the above lemmas in the 2D case.

3 A Linear Min-Max Problem in the Hough space

Solving the linear program is equivalent to finding the lowest point of the upper envelope of the set of planes defined by

$$z_k(x, y) = a_k x + b_k y + c \quad k = 1, 2, \ldots, n.$$

We saw that the upper envelope corresponds to the lower convex hull of the set of points $DP = \{(a_k, b_k, -c_k)\}_{k=1}^{n}$. It is now obvious that a solution to Problem 2, which is a point (x, y, t) on the upper envelope, corresponds to a plane in the dual space, which is defined by a face of $\mathcal{LH}(DP)$.

Consider an optimum for the target function of the linear problem, namely t_{opt}, and define the following plane which is parallel to the $x-y$ plane: $\pi(x, y) = 0 \cdot x + 0 \cdot y + 1 \cdot t_{opt}$. The dual to this plane, $\mathcal{H}(\pi)$, is a point on the z axis. Because the optimal solution p to Problem 2 is a point that is on π, its dual $\mathcal{H}(p)$ is a plane on which the point $\mathcal{H}(\pi)$ must be. This means that the dual to the plane that has the highest intersection with the $z = 0$ axis (and is part of $\mathcal{LH}(DP)$) is the solution to the linear program. Moreover, it is obvious that if there is a face of the lower convex hull that intersects the z axis, it has the highest intersecting plane. If there is no such face, either all of the points have a positive a_k, or all have a negative a_k, which means the problem is unbounded.

The last two statements suggest that solving our linear program is equivalent to finding the face of the lower convex hull that intersects the z axis. Unfortunately, we are unaware of an $\mathcal{O}(n)$ method to do so.

Figure 2 demonstrates the above lemmas in the 2D case.

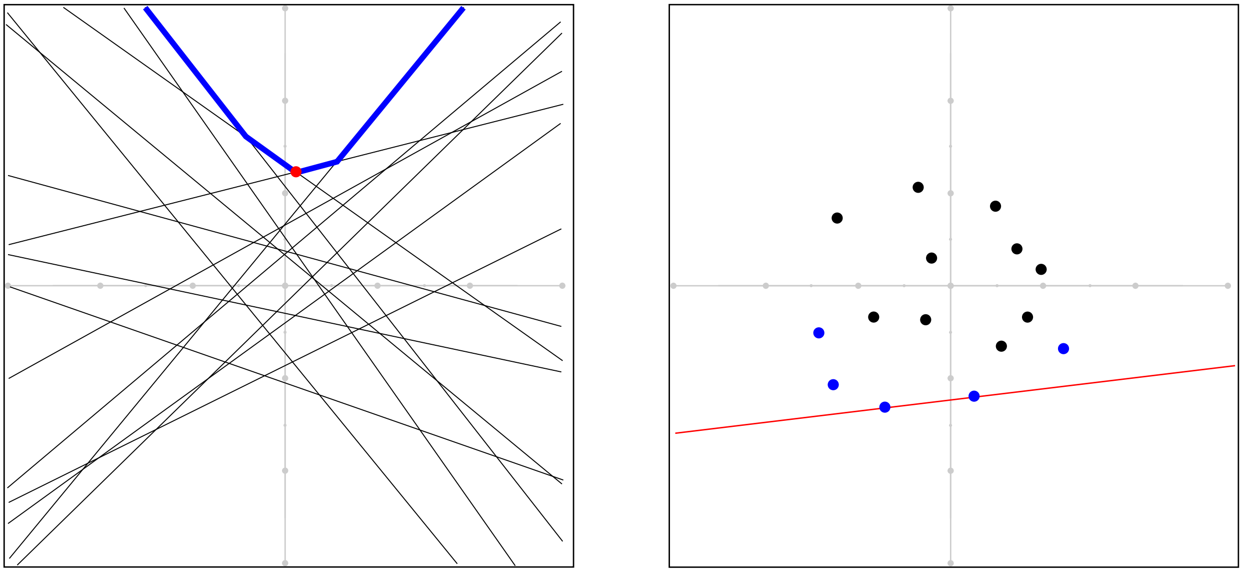

Figure 2: Left: a linear min-max problem, and its solution (red point). Right: duals of planes defining the problem, and the dual to the solution (red line).

4 The 2D Case: Solving Problem 2 in the Hough plane

Although the last section ends in a pessimistic note, this section provides an algorithm which in practice, solves 2D linear min-max problems much faster than CGAL's SOLVE_LINEAR_PROGRAM Fischer et al. (2011). We were unable, at the time of writing this work, to provide a complexity proof; however, experiments support the conjecture it is linear in the number of constraints. See Figure 3

To clarify: we remove the $y-$coordinate from our problem formulation and rename z into y, to obtain the following class of problems:

$$\begin{aligned} \text{minimize}_{x,t} \quad & t \\ \text{s.t} \quad & a_1 x + b_1 \leq t \\ & \vdots \\ & a_n x + b_n \leq t. \end{aligned}$$

The algorithm is as follows.

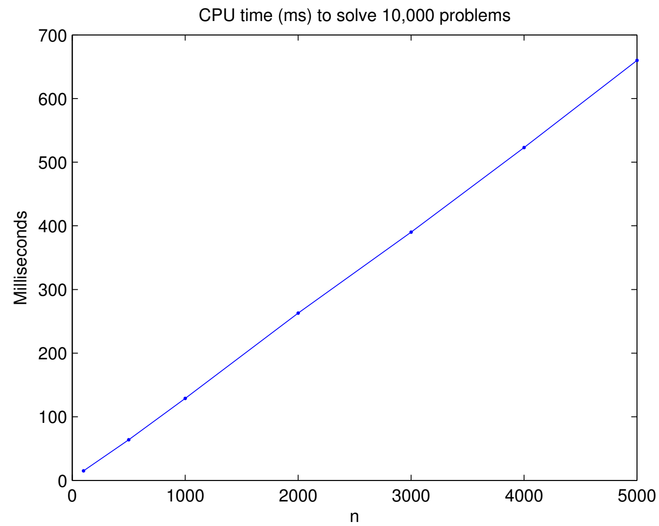

Figure 3: CPU time in milliseconds to solve 10,000 random problems with n constraints (our algorithm)

Algorithm 1 Solving in Hough plane (2D)

Input: A set of n constraints: $a_i x + b_i \leq t, \quad i \in [n]$
Output: Optimal primal point (x, t)

1. Transform constraints into a set of points DP
2. Partition DP into L (points with negative $x-$coordinate) and R (positive)
3. Pick some point $p_0 \in L$
4. Repeat until no change:
   (a) Given p_i in L (R), find a point p_{i+1} in R (L) with the largest clockwise (counter-clockwise) turn from p_i.
5. Compute the line that intersects the last two points p_k and p_{k+1}: $\ell(x) = mx - mp_k^{(x)} + p_k^{(y)}$ where $m = \left(p_{k+1}^{(y)} - p_k^{(y)}\right) / \left(p_{k+1}^{(x)} - p_k^{(x)}\right)$
6. Compute the dual to this plane: $\left(m, mp_k^{(x)} - p_k^{(y)}\right)$
7. Return either (6) or $-\infty$.

First, we should prove the algorithm takes a finite number of steps.

Claim 5. Step (4) in Algorithm 1 terminates.

Proof. Suppose we are at step $i+1$, and the last two points are p_i and p_{i+1}. Also assume without loss of generality that $p_i \in L$ and therefore $p_{i+1} \in R$. Next, either step (4a) terminates (we're done) or a new point $p_{i+2} \in L$ is selected. The criterion for the selection is that p_{i+2} constitutes a counter-clockwise turn from p_i, which is equivalent to to the line (p_{i+2}, p_{i+1}) having a larger inclination than the line (p_i, p_{i+1}). However, because both lines intersect p_{i+1}, and the second one has a larger inclination, its intersection with the y axis is smaller than that of the first one. This means that the $y-$intersection of any (p_i, p_{i+1}) is smaller than the $y-$intersection of any line (p_j, p_{j+1}) for all $i > j$. Because the number of points is finite, the minimum of $y-$intersections of lines defined by any pair of points in DP is finite, therefore step (4a) terminates. $\square$

We are left with showing that when step (4a) terminates, the points p_k and p_{k+1} are indeed part of $\mathcal{LH}(DP)$; the fact that the segment (p_k, p_{k+1}) intersects the y axis is obvious.

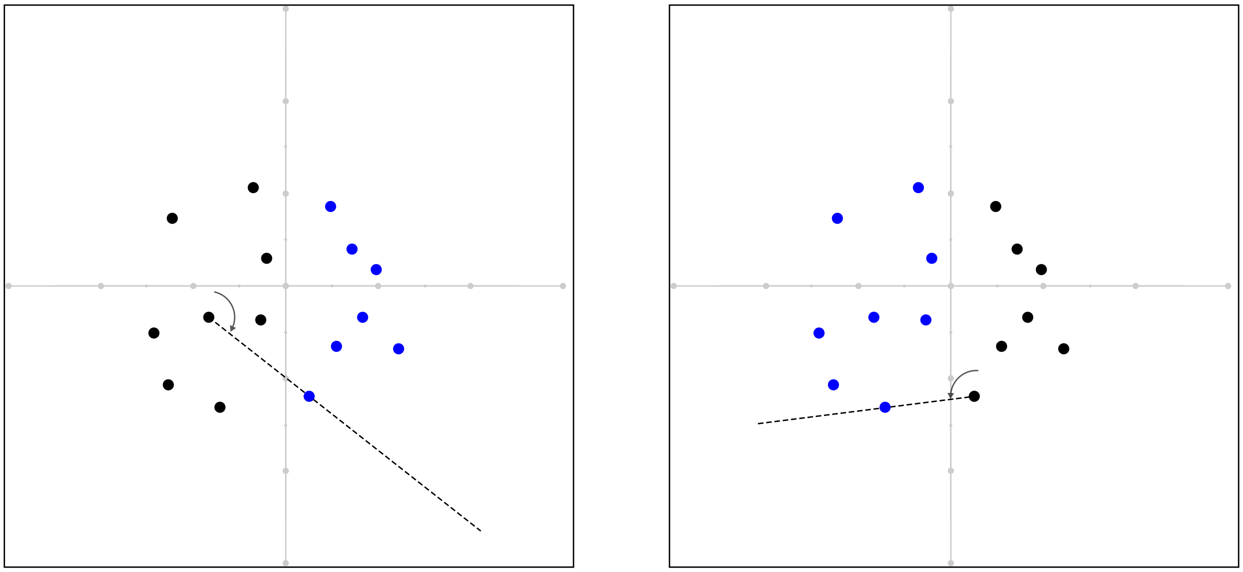

Figure 4: Two applications of step 4a

Proof. Assume without loss of generality that $p_k \in L$ and therefore $p_{k+1} \in R$. By step (4a) we know that no point in R has a clockwise turn from the line (p_k, p_{k+1}), or equivalently, that all the points in R are above this line. Similarly, no point in L is above this same line for symmetric reasons. This means this line supports DP (from below), which means it must be on its lower convex hull. □

Figure 4 demonstrates the algorithm.

4.1 Experimental Verification

We pit our algorithm against CGAL's linear programming solver Fischer et al. (2011). The constraints were drawn from a 2D Gaussian random variable $X \sim \left[N\left(E=0, \sigma^2=10\right)\right]^2$. As is evident in Figure 5 which depicts the ratio between running times of the two solvers, versus the number of constraints n, the new algorithm is about 10 times faster than CGAL's.

4.1.1 Implementation notes

Our solver only uses addition and multiplication and is highly-parallelizable. No divisions are used, which results in a faster and more stable solver.

CGAL's solver requires the use of so-cold exact types, such as rational numbers or arbitrary-precision floating point numbers; the use of such types is extremely slow at this time, because of the way CGAL uses them. Therefore, the solver was tricked to using simple machine double-precision floating point numbers. This resulted in numerical failure when the number of constraints n was greater than $\approx 10^5$. Our solver, however, is working even with this number of constraints. Moreover, the only numerically sensitive step in our algorithm is in Step (4a), which involves deciding if three points define a clockwise turn. This is done by computing $(p_1 - p_0) \wedge (p_2 - p_0) \overset{?}{>} 0$, which can be done by setting $\tilde{p}_i = p_i - p_0$ and then deciding if

$$\tilde{p}_1^x \cdot \tilde{p}_2^y \text{ is greater than } \tilde{p}_1^y \cdot \tilde{p}_2^x,$$

or equivalently,

$$\frac{\tilde{p}_1^x}{\tilde{p}_1^y} \text{ is greater than } \frac{\tilde{p}_2^x}{\tilde{p}_2^y}.$$

This comparison can be performed in exact by converting each fraction to a continued fraction, which is still much faster than using an arbitrary-precision floating point number.

A computer using Core 2 Quad (Q9400) @ 2.66 GHz and 6GB RAM was used for the benchmarking. 10,000 random problems were solved by each of the solvers, and their results compared.

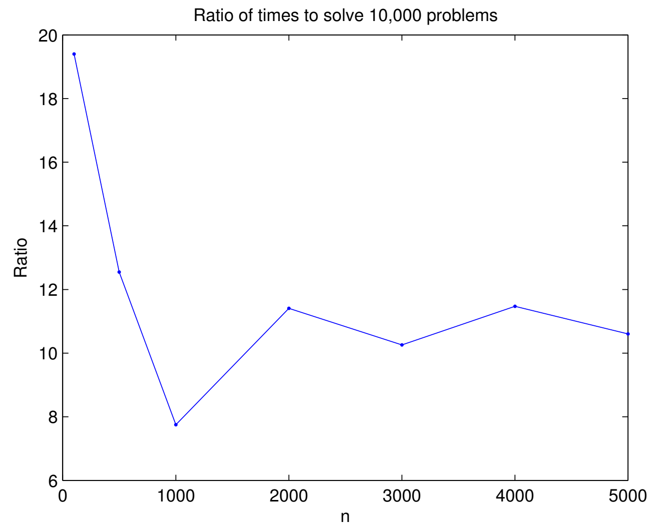

Figure 5: Ratio of time to solve 10,000 problems, vs. problem size (n)

5 The 3D Case: Discarding of Constraints

To again recede to a pessimistic note, we were unable to extend the 2D algorithm into 3D. However, modifying the problem at hand allows us to quickly and safely discard of some of the constraints. The modified problem has two more constraints: the solution point (x, y, t) must have its coordinates x and y between 0 and 1.

Problem 6.

$$\begin{aligned} \text{minimize}_{x,y,t} \quad & t \\ \text{s.t} \quad & a_1x + b_1y + c_1 \leq t \\ & \quad\vdots \\ & a_nx + b_ny + c_n \leq t \\ & x \in [0,1] \\ & y \in [0,1]\,. \end{aligned}$$

This problem is indeed very specific, but is in the core of the GMDS algorithm Bronstein et al. (2006) in the L_∞ norm, where x and y represent barycentric coordinates inside a triangle.

To reiterate, solving Problem 6 is equivalent to finding the face of the 3D lower convex hull of the points which are dual to the planes defined by the constraints, which intersects the z axis. In addition, if this (only) plane $z(x, y) = ax + by + z$ has either $a \notin [0, 1]$ and/or $b \notin [0, 1]$, the solution must be on the boundary of the feasible set.

In other words, it is possible to discard of all of the points in DP which support planes with either $a \notin [0, 1]$ and/or $b \notin [0, 1]$ and not change the solution, provided that the boundaries of the feasible set are checked. We note that a corresponds to the inclination in the x direction, and b to the inclination in the y direction.

To illustrate the points which can be discarded, consider a simplification of the problem to 2D. Figure 6 shows a set of points for which we should find a line that supports the rest of the points, from below, and has the highest $y-$intersect. Surely, we can discard of the points on the left of the lowest point, because these points are either interior, or support segments of $\mathcal{LH}(DP)$ which have negative inclinations. Also, points on the right of the lowest point, which define lines with inclination greater than 1 can also be discarded.

The last two statements are true in 2D, but not necessarily in 3D - removing points changes the convex hull, and there is danger that the new convex hull will introduce bogus solutions to the problem. We will see later that this is not the case, and removing the points is indeed safe.

The main result is detailed in Theorem 12, which is based on the following propositions.

Definition 7. A point $p = (p_x, p_y, p_z)$ is *behind* another point $q = (q_x, q_y, q_z)$ if $p_x < q_x$, $p_y < q_y$ and $p_z > q_z$.

Proposition 8. *Let $p^{min} \in DP$ be a point with a minimal $z-$coordinate, and let $p \in DP$ be point which is behind p^{min}. Then any plane defined by $n \cdot (q - p) = 0$ that:*

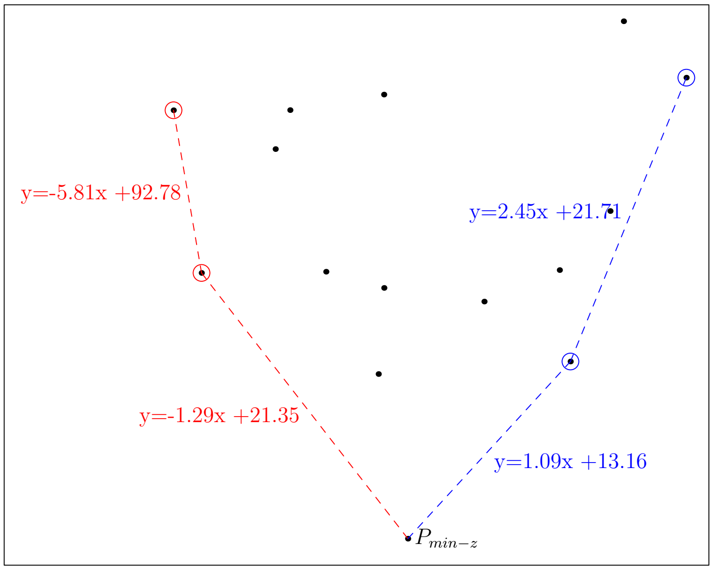

Figure 6: Points in red only support segments which have a negative inclination. Blue ones only support segments with inclinations greater than 1.

1. *Goes through p,*
2. *Supports $\mathcal{CH}(DP)$, that is, for any point $q \in DP$ we have $n \cdot (q - p) \geq 0$, and;*
3. *Has a positive $z-$coordinate n_z*

cannot be a solution to Problem 6 , because its dual point has negative x and/or y coordinate.

Proof. Apply Requirement (2) to the point $q = p^{min}$, for which $q - p$ equals $(+\alpha, +\beta, -\gamma)$ for some positive α, β and γ. The result is the relation $n_x\alpha + n_y\beta > n_z\gamma$, which forces either of n_x or n_y to be positive. Now, because (n_x, n_y, n_z) is a normal to a plane, the corresponding plane equation must be $n_z \cdot z(x, y) = (-n_x)x + (-n_y)y + c$ for some constant c. Because n_z is positive by Requirement (3), either of the coefficients of the plane must be negative, which means the dual to this plane cannot be a solution to Problem 6. □

We can state a similar result for points which are "too steep" with respect to p^{min}.

Definition 9. A point $p = (p_x, p_y, p_z)$ is *too steep with respect to* another point $q = (q_x, q_y, q_z)$ if $p \succ q$ (larger in all coordinates), and

$$\frac{p_z - q_z}{p_x - q_x} > 1 \text{ and } \frac{p_z - q_z}{p_y - q_y} > 1.$$

Proposition 10. *Let $p^{min} \in DP$ be a point with a minimal $z-$coordinate, and let $p \in DP$ be a point which is* too steep with respect to *p^{min}. That is, with all coordinates larger than those of p^{min} such that the vector $p^{min} - p = (-\alpha, -\beta, -\gamma)$, α, β and γ positive, satisfies*

$$\alpha < \gamma \text{ and } \beta < \gamma.$$

Then any plane defined by $n \cdot (q - p) = 0$ that:

1. *Goes through p,*
2. *Supports $\mathcal{CH}(DP)$, that is, for any point $q \in DP$ we have $n \cdot (q - p) \geq 0$, and;*
3. *Has a positive $z-$coordinate n_z*

cannot be a solution to Problem 6, because its dual point has its x and/or y coordinate larger than 1.

Proof. In a similar way to the proof of Proposition 8, we obtain

$$(n_x, n_y, n_z) \cdot (-\alpha, -\beta, -\gamma) = (-n_x)\alpha + (-n_y)\beta - n_z\gamma,$$

which implies

$$\left(-\frac{n_x}{n_z}\right)\alpha + \left(-\frac{n_y}{n_z}\right)\beta > \gamma.$$

We can normalize n such that n_z will be 1, without changing its sign, because n_z is positive by Requirement (3):

$$(-n_x)\frac{\alpha}{\gamma} + (-n_y)\frac{\beta}{\gamma} > 1.$$

Now, if both α/γ and β/γ are smaller than 1, n_x and n_y cannot be both larger than -1. Noting that the explicit plane equation which the normal vector n (with $n_z = 1$) induces is $z(x, y) = (-n_x)x + (-n_y)y + c$ for some c, leads to the conclusion that at least one of the coefficients of the plane must be larger than 1, and therefore its dual point cannot be a solution to the linear program. □

We state a simple characterization of lower convex hulls :

Lemma 11. *Any plane π defined by $n \cdot (q - p) = 0$, which supports $\mathcal{LH}(DP)$ must have a positive $z-$coordinate of n , n_z.*

Proof. Since π is part of the lower convex hull, there must be a pair of points q_1 and q_2 such that q_1 is on π, and $q_2 = q_1 + \epsilon(0, 0, 1)$; therefore, the condition $n \cdot (q_1 - q_2) > 0$ forces $n_z > 0$. □

We are now ready to show that we can safely discard of a class of constraints.

Theorem 12. *Discarding of points which are behind p^{min} (as defined in Proposition 8), or are too steep (as defined in Proposition 10) leads to a min-max problem which is equivalent to Problem 6.*

Proof. We will prove for points which are behind p^{min}; the proof for points which are too steep is almost identical, and is left for the reader.

Let p be a point which is behind p^{min}. First, any face which p supports and is part of $\mathcal{LH}(DP)$ cannot be a solution, therefore removing p does not change the solution to Problem 6 (apply Lemma 11 and Proposition 8). However, there still remains a possibility that removing p creates new faces in $\mathcal{LH}(DP)$ which results in a bogus solution to Problem 6. We now show it is not the case.

We consider the incremental convex hull construction algorithm detailed in de Berg (2000). Suppose we have $A = \mathcal{LH}(DP \backslash p)$, and would like to construct $B = \mathcal{LH}(DP)$. This is done by finding all the faces of A which are *visible* to p, that is, all faces which separate p from $DP \backslash p$. These faces are then removed, and the hole is filled using faces that p supports.

A convex hull of a set of points P is unique; one way to see this is to recall one definition of $\mathcal{CH}(P)$ - the intersection of all convex sets that contain P. This uniqueness implies that the faces that are added to the convex hull, when we remove p, are the faces that would have been removed if we constructed B from A using the aforementioned algorithm. This characterizes the faces that are added to A when we remove p - they all define planes which separate p from $DP \backslash p$. See Figure 7.

Formally, these planes are defined by some normal n and a point p', and have $n \cdot (q - p') \geq 0$ for all $q \in DP \backslash p$ but $n \cdot (p - p') \leq 0$. Such a plane, when translated so that its passes through p, is defined by $n \cdot (q - p) = 0$. We note that for any $q \in DP \backslash p$, we have

$$\begin{aligned} n \cdot (q - p) &= n \cdot (q - p + p' - p') \\ &= n \cdot (q - p') + (-n \cdot (p - p')) \\ &\geq 0, \end{aligned}$$

which means the translated plane goes through p, supports $\mathcal{CH}(DP)$ and has a positive n_z component (we assumed the original face belonged to $\mathcal{LH}(DP)$). This means it satisfies the requirements of Proposition 8, and therefore this plane (translated or not) cannot be a solution to Problem 6, which concludes the proof. □

With regard to the computational cost of these purgings, we note that they are highly-parallelizable, in addition to being a single-pass over the points.

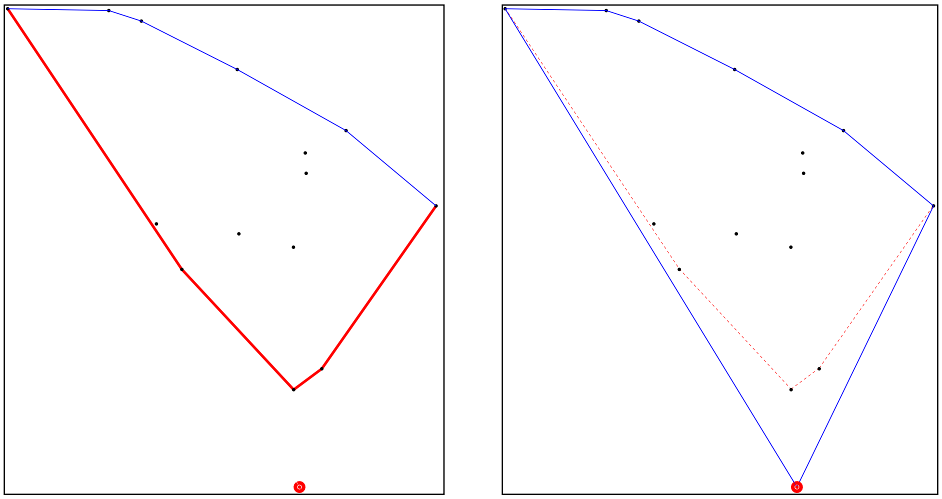

Figure 7: Incremental convex hull construction. Left: before adding the marked point, with visible faces emphasized. Right: after the addition. Note that removing the marked point from the convex hull on the right image, would result in the convex hull that appears in the left image.

References

Alexander M. Bronstein, Michael M. Bronstein, and Ron Kimmel. Generalized multidimensional scaling: a framework for isometry-invariant partial surface matching. *Proc. National Academy of Sciences (PNAS)*, Volume 103/5:1168–1172, January 2006.

Mark de Berg. *Computational geometry: algorithms and applications*. Springer, 2000. ISBN 9783540656203. URL `http://books.google.com/books?id=C8zaAWuOIOcC`.

Kaspar Fischer, Bernd Gärtner, Sven Schönherr, and Frans Wessendorp. Linear and quadratic programming solver. In *CGAL User and Reference Manual*. CGAL Editorial Board, 3.8 edition, 2011. http //www.cgal.org/Manual/3.8/doc_html/cgal_manual/packages.html#Pkg QPSolver.

Paul V. C. Hough. Method and means for recognizing complex patterns, March 1962. URL `http://www.google.com/patents?vid=3069654`.

Nimrod Megiddo. Linear programming in linear time when the dimension is fixed. *Journal of The ACM*, 31:114–127, 1984. doi: 10.1145/2422.322418.